Python и Ruby под нагрузкой:
самый низкий уровень

Григорий Петров





/evrone.

Давайте использовать социальные сети:

grigoryvp@evrone.com

t.me/grigoryvp

fb.com/grigoryvp

vk.com/grigoryvp

github.com/grigoryvp

twitter.com/grigoryvp

instagram.com/grigoryvp

Слайды → bit.ly/hilpyrb

Соцсети → @grigoryvp

Имя → Григорий Петров



/evrone.

Что сейчас будет?

```
{
    speaker_name: "Григорий Петров",
    skill_list: ["Python", "Ruby", "JS", "Go", "Rust", "Cpp"],
    skill_years: 20,
    job_title: "DevRel",
    talk_minutes: 30,
    questions_at: TALK_END,
}
```




/evrone.

Почему Python и Ruby?!



/evrone.

Почему Python и Ruby?!





Почему Python и Ruby?!

TESLA MODEL 3: 85.62%

DEEP LEARNING FOR COMPUTER VISION WITH PYTHON

Dr. Adrian Rosebrock

pyimagesearch





Почему Python и Ruby?!

DEEP LEARNING FOR COMPUTER VISION WITH PYTHON

TESLA MODEL 3: 85.62%

Dr. Adrian Rosebrock

pyimagesearch

BACANCY TECHNOLOGY

How Full Stack RoR Developer Can Help You Get Your Startup Off the Ground

- CIO Edition

www.bacancytechnology.com



Этим кто-нибудь пользуется?



/evrone.

Этим кто-нибудь пользуется?





/evrone.

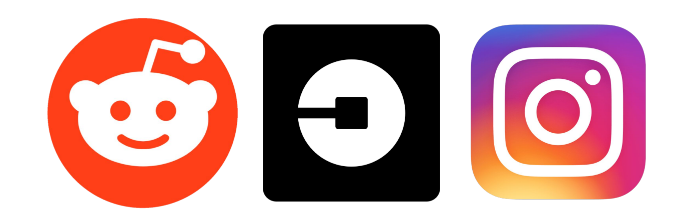
Этим кто-нибудь пользуется?





/evrone.

Этим кто-нибудь пользуется?







Этим кто-нибудь пользуется?





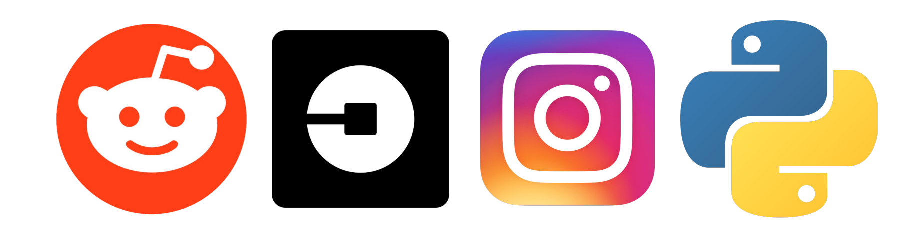
Этим кто-нибудь пользуется?







Этим кто-нибудь пользуется?







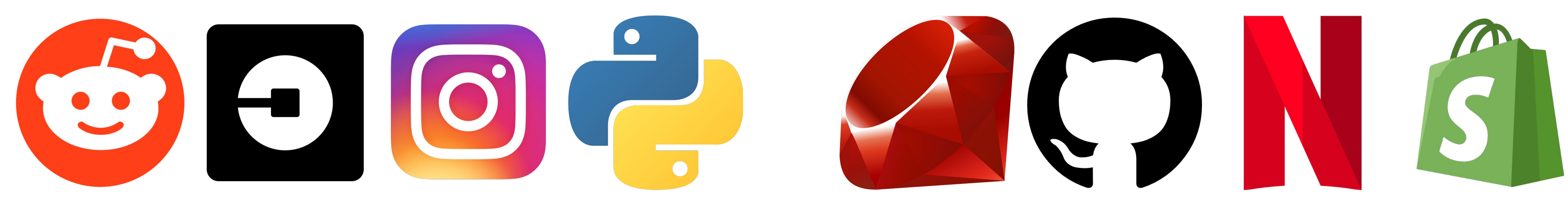
Этим кто-нибудь пользуется?



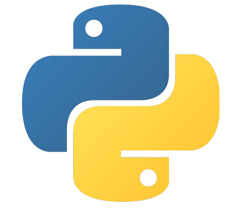
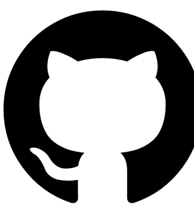




Этим кто-нибудь пользуется?









Этим кто-нибудь пользуется?

https://charliereese.ca/article/top-50-y-combinator-tech-startups

1. Top 50 Software Startups:

Company	Latest val ($MM)	Initial back-end language(s)	DataSci / LowLv
Stripe: Payment / economic infrastructure for internet	36,000 source	Ruby source	N / N
Cruise: Building self-driving car tech	19,000 source	C++, Python source, source	Y / N
Airbnb: Marketplace to rent someone's room	18,000 source	Ruby source	N / N
DoorDash: Food delivery	16,000 source	Python source	N / N
Twitch: Gaming video platform / community	15,000 source	C++, Ruby source (founded before Go)	N / Y
Instacart: Grocery pick-up / delivery	13,700 source	Ruby source	N / N
Dropbox: File hosting / syncing	8000 (market cap @ Aug 2020) source	Python source	N / Y
Coinbase: Cryptocurrency exchange	8,000 source	Ruby source	N / N



/evrone.

О чем этот доклад?



/evrone.

О чем этот доклад?

Мейнстрим



/evrone.

О чем этот доклад?

Мейнстрим

- Не competitive development.



/evrone.

О чем этот доклад?

Мейнстрим

✓ Не competitive development.
➜ Популярные app servers.





О чем этот доклад?

Мейнстрим

✓ Не competitive development.

✓ Популярные app servers.

➜ HTTP/1.1



О чем этот доклад?

Мейнстрим

Python и Ruby

- ✓ Не competitive development.
- ✓ Популярные app servers.
- ✓ HTTP/1.1



/evrone.

О чем этот доклад?

Мейнстрим

- ✓ Не competitive development.
- ✓ Популярные app servers.
- ✓ HTTP/1.1

Python и Ruby

- ➜ Одна нода приложения.

/evrone.

О чем этот доклад?

Мейнстрим

- ✓ Не competitive development.
- ✓ Популярные app servers.
- ✓ HTTP/1.1

Python и Ruby

- ➜ Одна нода приложения.

/evrone.

О чем этот доклад?

Мейнстрим

- ✓ Не competitive development.
- ✓ Популярные app servers.
- ✓ HTTP/1.1

Python и Ruby

- ✓ Одна нода приложения.
- ➜ Процессы, потоки, GIL.

/evrone.

О чем этот доклад?

Мейнстрим

✓ Не competitive development.
✓ Популярные app servers.
✓ HTTP/1.1

Python и Ruby

✓ Одна нода приложения.
✓ Процессы, потоки, GIL.
➜ Кэширование ускоряет всё.



О чем этот доклад?

Мейнстрим

- ✓ Не competitive development.
- ✓ Популярные app servers.
- ✓ HTTP/1.1

Python и Ruby

- ✓ Одна нода приложения.
- ✓ Процессы, потоки, GIL.
- ✓ Кэширование ускоряет всё.
- ➜ База тормозит всех.



Python и Ruby начинаются с nginx

NGINX





Python и Ruby начинаются с nginx

➜ Историческая защита мягкого подбрюшья "application servers".





Python и Ruby начинаются с nginx

✓ Историческая защита мягкого подбрюшья "application servers".

➜ Сертификаты, HTTP/2, кэширование.





Python и Ruby начинаются с nginx

- ✓ Историческая защита мягкого подбрюшья "application servers".
- ➜ Сертификаты, HTTP/2, кэширование.





Python и Ruby начинаются с nginx

✓ Историческая защита мягкого подбрюшья "application servers".

➜ Сертификаты, HTTP/2, кэширование.





Python и Ruby начинаются с nginx

- ✓ Историческая защита мягкого подбрюшья "application servers".
- ✓ Сертификаты, HTTP/2, кэширование.
- ➜ HTTP или binary коммуникация с application server.



/evrone.

Application server: пастух стада питонов







Application server: пастух стада питонов

Rainbows!

Mongrel

FastCGI

Other 1%

Unicorn 16%

Passenger (mod_rails) 21%

Puma 61%

0 200 400 600 800 1000 1200 1400

Total





Application server: пастух стада питонов

WSGI Servers

Nginx + uWSGI

This is one of fast and realistic way to serve Python web application.

Use unix domain socket between nginx and uWSGI to avoid additional TCP/IP overhead.





Application server: пастух стада питонов

➜ Запускает Python/Ruby, загружает код, следит за выполнением.





Application server: пастух стада питонов

✓ Запускает Python/Ruby, загружает код, следит за выполнением.

➜ Контроль количества процессов и потоков.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ➜ Контроль и ограничение ресурсов: память, процессор.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ➜ Перезапуск зависших, протекших, тормозящих воркеров.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ➜ Обновление и перезапуск без обрыва соединений.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ➜ Метрики и логи в разные бэкенды.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ➜ API для плагинов.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ➜ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ➜ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ➜ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ➜ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ➜ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ➜ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ✓ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.
- ➜ Собственный уровень роутинга и кэширования.





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ✓ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.
- ➜ Собственный уровень роутинга и кэширования.

много всего





Application server: пастух стада питонов

- ✓ Запускает Python/Ruby, загружает код, следит за выполнением.
- ✓ Контроль количества процессов и потоков.
- ✓ Контроль и ограничение ресурсов: память, процессор.
- ✓ Перезапуск зависших, протекших, тормозящих воркеров.
- ✓ Обновление и перезапуск без обрыва соединений.
- ✓ Метрики и логи в разные бэкенды.
- ✓ API для плагинов.
- ✓ Фоновые задачи, очереди, таймеры, локи, RPC, WebSockets.
- ➜ Собственный уровень роутинга и кэширования.

много всего





Процессы, потоки, GIL и GC

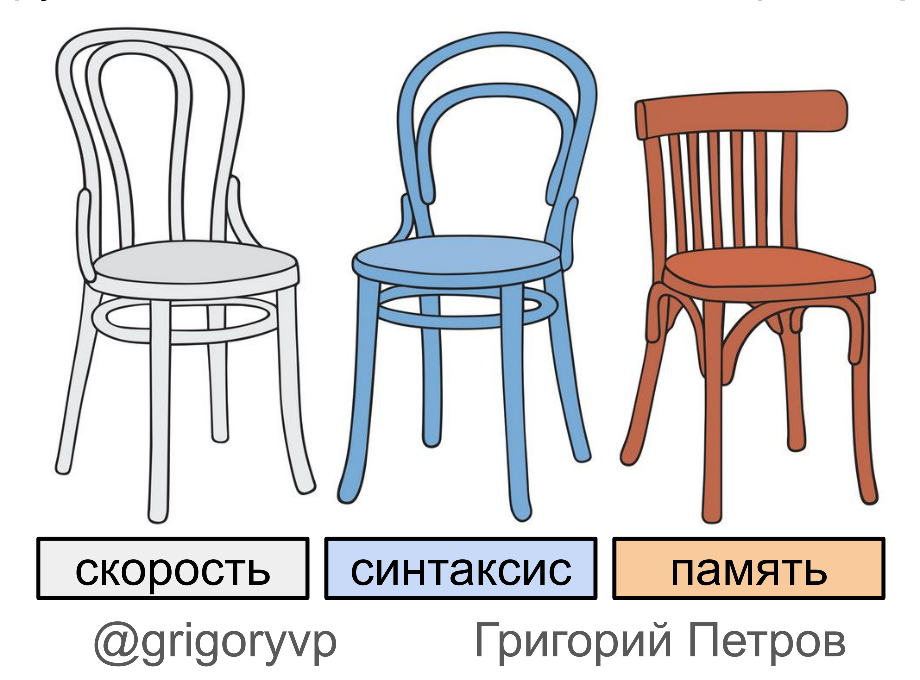
➜ Python, Ruby, PHP: GIL для простого управления памятью.





Процессы, потоки, GIL и GC

- ✓ Python, Ruby, PHP: GIL для простого управления памятью.
- ➜ Это цена, которую языки платят за синтаксис и расширяемость.







Процессы, потоки, GIL и GC

- ✓ Python, Ruby, PHP: GIL для простого управления памятью.
- ✓ Это цена, которую языки платят за синтаксис и расширяемость.
- ➜ Расширения на C "поднимают" GIL и многопоточны.





Процессы, потоки, GIL и GC

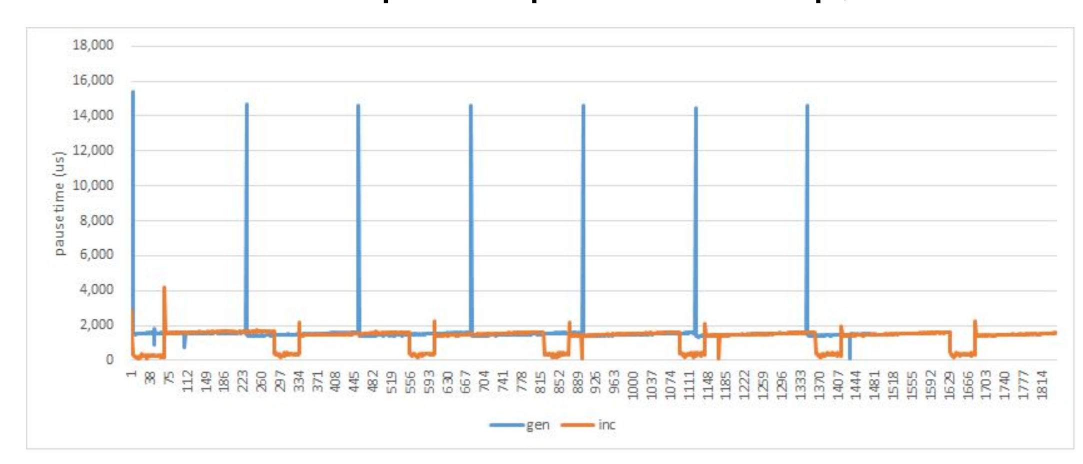
- ✓ Python, Ruby, PHP: GIL для простого управления памятью.
- ✓ Это цена, которую языки платят за синтаксис и расширяемость.
- ✓ Расширения на C "поднимают" GIL и многопоточны.
- ➜ Mark-and-sweep замораживает мир, но его можно отключить.





Процессы, потоки, GIL и GC

- ✓ Python, Ruby, PHP: GIL для простого управления памятью.
- ✓ Это цена, которую языки платят за синтаксис и расширяемость.
- ✓ Расширения на C "поднимают" GIL и многопоточны.
- ➜ Mark-and-sweep замораживает мир, но его можно отключить.

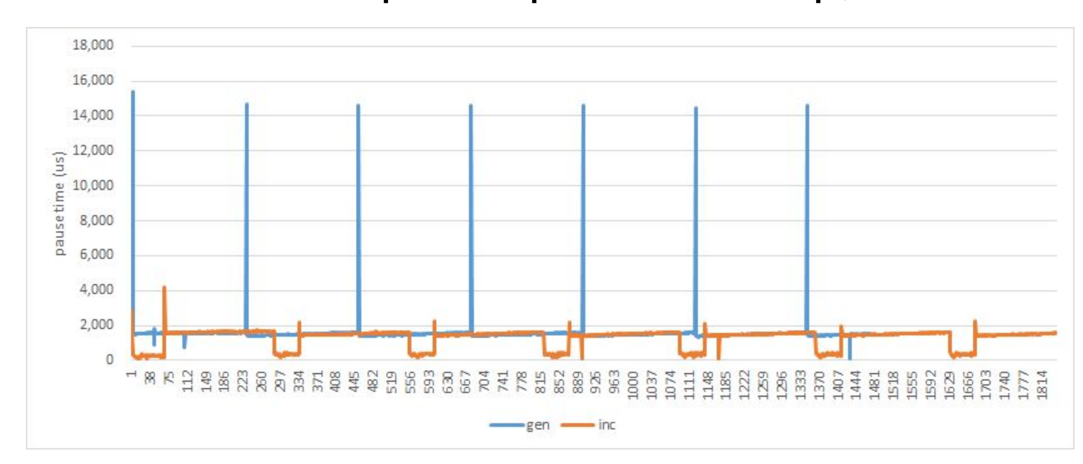






Процессы, потоки, GIL и GC

- ✓ Python, Ruby, PHP: GIL для простого управления памятью.
- ✓ Это цена, которую языки платят за синтаксис и расширяемость.
- ✓ Расширения на C "поднимают" GIL и многопоточны.
- ➜ Mark-and-sweep замораживает мир, но его можно отключить.







Процессы, потоки, GIL и GC

- ✓ Python, Ruby, PHP: GIL для простого управления памятью.
- ✓ Это цена, которую языки платят за синтаксис и расширяемость.
- ✓ Расширения на C "поднимают" GIL и многопоточны.
- ✓ Mark-and-sweep замораживает мир, но его можно отключить.
- ➜ Мейнстрим: процессы для параллелизма, потоки для асинхронности.



/evrone.

Python



/evrone.

Python

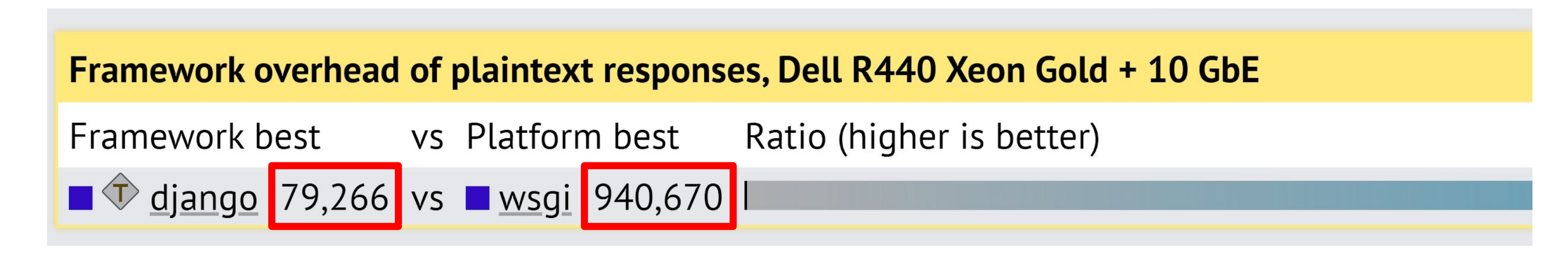
- Компилятор => байткод => VM для памяти и семантики.





Python

- ✓ Компилятор => байткод => VM для памяти и семантики.
- ➜ Bare Python / Django = 10 / 1







Python

- ✓ Компилятор => байткод => VM для памяти и семантики.
- ✓ Bare Python / Django = 10 / 1
- ➜ Почему тормозит фреймворк?

/evrone.

Фреймворк: Django

- ORM: migration, validation, auth, storage.





Фреймворк: Django

✓ ORM: migration, validation, auth, storage.
➜ Routing: controllers, redirects, API, DSL.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ➜ Rendering: templates, composition, assets.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ➜ Middleware API for plugins.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ➜ Cache.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ✓ Cache.
- ➜ Logging.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ✓ Cache.
- ✓ Logging.
- ➜ Testing.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ✓ Cache.
- ✓ Logging.
- ✓ Testing.
- ➜ Sessions, auth, forms, security, cfg, notify, email, files, i18n.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ✓ Cache.
- ✓ Logging.
- ✓ Testing.
- ✓ Sessions, auth, forms, security, cfg, notify, email, files, i18n.
- ➜ CLI: scaffolding, testing, dev server, plugin commands.





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ✓ Cache.
- ✓ Logging.
- ✓ Testing.
- ✓ Sessions, auth, forms, security, cfg, notify, email, files, i18n.
- ➜ CLI: scaffolding, testing, dev server, plugin commands.

много всего





Фреймворк: Django

- ✓ ORM: migration, validation, auth, storage.
- ✓ Routing: controllers, redirects, API, DSL.
- ✓ Rendering: templates, composition, assets.
- ✓ Middleware API for plugins.
- ✓ Cache.
- ✓ Logging.
- ✓ Testing.
- ✓ Sessions, auth, forms, security, cfg, notify, email, files, i18n.
- ➜ CLI: scaffolding, testing, dev server, plugin commands.

много всего



Ruby



Ruby

➜ Компилятор => байткод => VM для памяти и семантики.





Ruby

✓ Компилятор => байткод => VM для памяти и семантики.

➜ Bare Ruby / Rails = 10 / 1

Best plaintext responses per second, Dell R440 Xeon Gold + 10 GbE (10 tests)

Rnk	Framework	Best performance (higher is better)		
3	rack-sequel	218,754	45.1%	(3.1%)
7	rails	20,445	4.2%	(0.3%)





Ruby

✓ Компилятор => байткод => VM для памяти и семантики.
✓ Bare Ruby / Rails = 10 / 1
➜ Фреймворк тормозит по тем же причинам.





Ruby

- ✓ Компилятор => байткод => VM для памяти и семантики.
- ✓ Bare Ruby / Rails = 10 / 1
- ✓ Фреймворк тормозит по тем же причинам.
- ➜ JIT не помогает.



Выводы



/evrone.

Выводы

- Бизнес-логику можно писать на чем угодно.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ➜ CPU-bound масштабировался и масштабируется процессами.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ➜ База, память, сеть, диск, проц уже оптимизированы за нас.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ➜ Современный стек сложен, может "тормозить" в странных местах.



Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ➜ Современный стек сложен, может "тормозить" в странных местах.



Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ➜ Современный стек сложен, может "тормозить" в странных местах.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ✓ Современный стек сложен, может "тормозить" в странных местах.
- ➜ Во все остальные места можно вставить Rust.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ✓ Современный стек сложен, может "тормозить" в странных местах.
- ✓ Во все остальные места можно вставить Rust.
- ➜ Basecamp тратит на Ruby 15% Ops-бюджета.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ✓ Современный стек сложен, может "тормозить" в странных местах.
- ✓ Во все остальные места можно вставить Rust.
- ✓ Basecamp тратит на Ruby 15% Ops-бюджета.
- ➜ Lyft платит $0.14 AWS за каждую поездку.





Выводы

- ✓ Бизнес-логику можно писать на чем угодно.
- ✓ CPU-bound масштабировался и масштабируется процессами.
- ✓ База, память, сеть, диск, проц уже оптимизированы за нас.
- ✓ Современный стек сложен, может "тормозить" в странных местах.
- ✓ Во все остальные места можно вставить Rust.
- ✓ Basecamp тратит на Ruby 15% Ops-бюджета.
- ➜ Lyft платит $0.14 AWS за каждую поездку.



/evrone.

Это всё!

Вопросы?

